UNIX® Network Programming

W. Richard Stevens
Health Systems International

PRENTICE HALL
Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging-in-Publication Data

Stevens, W. Richard.
UNIX network programming / W. Richard Stevens.
p. cm.
Includes bibliographical references.
ISBN 0-13-949876-1
1. UNIX (Computer operating system) 2. Computer networks.
I. Title.
QA76.76.063S755 1990
005.7'1--dc20

89-25576
CIP

To Sally, Bill, and Ellen.

Editorial/production supervision: **Brendan M. Stewart**
Cover design: **Lundgren Graphics Ltd.**
Manufacturing buyer: **Ray Sintel**

Prentice Hall Software Series
Brian W. Kernighan, Advisor

A Division of Simon & Schuster
Englewood Cliffs, NJ 07632

UNIX® is a registered trademark of AT&T.

Printed in the United States of America
10 9 8 7 6 5 4

ISBN 0-13-949876-1

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Simon & Schuster Asia Pte. Ltd., *Singapore*
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

```
        int     pw_uid;         /* user-ID */
        int     pw_gid;         /* group-ID */
        int     pw_quota;       /* BSD-only; not used */
        char    *pw_age;        /* System V-only; password age */
        char    *pw_comment;    /* not used */
        char    *pw_gecos;      /* miscellany */
        char    *pw_dir;        /* login-directory */
        char    *pw_shell;      /* shell */
};
```

(The order of the elements in this structure might be different on your Unix system, but their names are as shown above.) Both the functions shown above return either a pointer to a `passwd` structure that has been filled in with the values from the appropriate entry in the `/etc/passwd` file, or `NULL` if a matching entry is not found. The `getpwuid` function searches for a matching user ID while the `getpwnam` function searches for a matching login name.

We'll encounter these two functions in the later chapters that cover line printer access, remote command execution, and remote login. The `getpwnam` function is used, for example, when a login name is passed between a client and server. The reason for using the name, instead of the user ID, is that your user ID might be different on the different systems on which you have a valid account. Therefore, the client has to obtain your login name by

```
struct passwd   *pwd;

pwd = getpwuid( getuid() );
```

The server then uses `getpwnam` to turn the name into your user ID.

Shadow Passwords

System V Release 3.2 introduced *shadow passwords*. This feature stores the encrypted passwords in a separate file, `/etc/shadow` and the *encrypted-password* field of the `/etc/passwd` file is set to an asterisk. The new shadow file, `/etc/shadow`, is set so that only the superuser can read the file and the original file, `/etc/passwd`, remains readable by anyone.

The problem with the original password file scheme is that even with a one-way encryption algorithm for the password field, intruders were taking copies of the `/etc/passwd` file and using common words as guesses. Since many users set their passwords to common words (their family names, common computer terms, common words backwards, and the like) a brute force search would often yield numerous valid passwords.

Another feature that was introduced with shadow passwords is *password aging*. This allows the system administrator to specify both the minimum and maximum number of days between password changes for a user.

of the process to be the group ID of the owner of this file." A program with this special flag is said to be a *set-group-ID* program. Like the set-user-ID feature, this provides additional permissions to users while the set-group-ID program is being executed.

We'll encounter all four of these IDs when we discuss the security aspects of network programming.

Superuser

User ID zero is special—it identifies the *superuser*. The login name for the superuser is usually *root*. The superuser is allowed unrestricted access to files and additional permissions over other processes. For example, the superuser can terminate any other process on the system, a privilege not available to other user IDs.

A process with an effective user ID of zero is termed a superuser process. When a system function is said to be "restricted to the superuser" this means that the process must have an effective user ID of zero to do the specified operation.

Password File

Each line in the `/etc/passwd` file has the following format:

login-name:*encrypted-password*:*user-ID*:*group-ID*:*miscellany*:*login-directory*:*shell*

There are seven fields, separated from each other by colons. The *login-name* is the name you enter in response to the `login:` prompt when logging on to the system. This field is sometimes called the *user name*. The *encrypted-password* field can be empty, in which case you are not prompted for a password. Since the passwords in this file are encrypted, this file is always readable by anyone. The *user-ID* and *group-ID* fields are the numeric values described above. The *login-directory* specifies your initial current working directory. The *shell* field specifies the pathname of a program that is invoked when you login. All these fields are described in more detail later in this chapter. Two typical entries are

```
root:x7f1mVqMxGl4g:0:10:The Superuser:/:/bin/ksh
stevens:u0ud5eOq2MpaZ:224:5:Richard Stevens:/usr1/stevens:/bin/ksh
```

The standard C library provides two functions to search the `/etc/passwd` file, looking for a matching user ID or login name.

```
#include <pwd.h>

struct passwd *getpwuid(int uid);

struct passwd *getpwnam(char *name);
```

The include file `<pwd.h>` defines a structure with the following elements:

```
struct passwd {
    char    *pw_name;       /* login-name */
    char    *pw_passwd;     /* encrypted-password */
```